

Office européen des brevets Europäisches Patentamt European Patent Office



EP 1 111 483 A1

Ξ

EUROPEAN PATENT APPLICATION published in accordance with Art. 158(3) EPC

**2** 

27.06.2001 Bulletin 2001/26 (43) Date of publication:

(51) Int Cl.7: G05B 19/02

(21) Application number: 99925309.9

(22) Date of filing: 14.06.1999

PCT/JP99/03146

(86) International application number:

WO 00/02106 (13.01.2000 Gazette 2000/02) (87) International publication number:

> (84) Designated Contracting States: DE FR GB

(71) Applicant: TOYOTA JIDOSHA KABUSHIKI

Alchi-ken 471-8571 (JP)

(30) Priority: 06.07.1998 JP 19081598

Leson, Thomas Johannes Alois, Dipt.-Ing. et af Toyota-shi, Alchi-ken 471-8571 (JP) Tiedtke-Bühling-Kinne & Partner, (72) Inventor: Adachi, Noriyasu Representative: Patentanwälte (74)

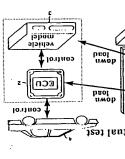
80336 München (DE) Bavarlaring 4

METHOD AND APPARATUS FOR ASSISTING DEVELOPMENT OF PROGRAM FOR VEHICLE

3

cally generates a vehicle C code from control is downloaded in a vehicle ECU (2). The control of a The man-hour required to develop a vehicle control program is reduced. A program generator (1) auand in the form of a state flow chart. The vehicle C code controlled system, which is a vehicle model device (3), erator (1) and debugging is performed. Since the vehicle lons is used, the debugging is easy and reliable. Manual specifications inputted in the form of a data flowchar by the vehicle ECU (2) is monitored by the program gen-C code automatically generated from control specificacoding is obviated. Integer information is added to a symbol block of a data flowchart, and therefore a vehicle C code of integer logic can be automatically generated using the integer Information.

Fig. 1



con (1.0] Specification nctual test uapi

Printed by Joune, 75001 PARIS (FR)

EP 1 111 483 A1

EP 1 111 483 A1

2

Description

develop highly reliable programs in a shorter period.

DISCLOSURE OF INVENTION

*TECHNICAL FIELD* 

The present invention relates to a method and apparatus for assisting development of a vehicle-use program, and in particular to an apparatus and method which improves development efficiency and reliability of a developed program.

BACKGROUND ART

cilitating control system design. Computer software for al programming technique have been developed for fasimutation and programming functions is now common-0002] Tools for integrated environment utilizing visu-

ly used to develop vehicle control systems.

tions are then executed according to the charts pre-pared, and the logic is debugged with reference to the [0003] When a computer operator working on system using a simulation function, a specification incorporating the concepts input by the user is shown on a computer display in the form of a data or state flowchart. Simuladevelopment inputs specification data into his computer

engthy with many steps and thus not suitable for use Such simulation software has a function of automatic generation of a program code, such as C language code, based on a chart prepared. C language code prepared in this manner is, however, generally intact in vehicle control, which requires severe restriclions on memory and an execution speed. Moreover, such C code is very unfriendly because computer-determined variable names are attached to parts in the [0004]

The situation being such, many users choose to manually generate vehicle-use C code suitable for venicle electric control units (vehicle ECU) while referring to the data and state flowcharts. In particular, dedicated codes having integer logic are generated. These codes differ from general C codes and enable high speed processing using a smaller memory in view of data bit number. This manually generated vehicle-use C code is Conventionally, as described above, the specfication simulation process and the program coding bugging vehicle-use code, it is difficult to ascertain whether the problem originated from erroneous coding or an improper original specification. Due to the difficulty ity is increased, which in tum prolongs the development downloaded to a vehicle-mounted ECU, and debugged. process are separate, and each requires a debug operation. Therefore, should a problem be found when dein coding and debugging operation, the number of process steps necessary to ensure control program reliabilperiod and increases development costs. [0000] ... ....

The present invention was conceived in light ratus for assisting vehicle-use program development to of the above, and aims to provide a method and appa-

ng to the present invention, there is provided a methoo for assisting vehicle program development. In this meth od, a vehicle control program is generated using a pro gram generator having a function of generating a vehicle-use code from a control specification Input. The gen erated vehicle control program is downloaded to a vehicle ECU, which in turn executes the vehicle contro In order to achieve the above objects, accord program for debugging the vehicle control program.

to occur. Therefore, efficient debug operation can be achieved for smaller labor. Moreover, in the person ECU. Because the original control specification is in common with the vehicle-use code, bugs are less likely hours required for coding operation can be significantly [0009] As described above, according to the presen invention, a vehicle-use code is generated directly from a control specification, and downloaded to a vehicle 8 5

bly performed in a program generator which inspects a result of execution of the vehicle control program by the vehicle ECU. Use of a program generator having a con-[0010] Debug operation at the debug step is prefera trol specification can facilitate a debut operation.

hicle program development, comprising a chart generation function of generating a data flowchart and a state [0011] According to another aspect of the present invention, there is provided an apparatus for assisting vebased on the generated charts, a vehicle-use code for ment, automatic generation of vehicle-use codes are achieved through generation of integer logic codes flowchart indicative of a vehicle control specification, a vehicle control program having an integer logic to be memory size and an execution speed. In this embodiand a program code generation function of generating processed by a vehicle ECU. Use of integer logic in the vehicle ECU is preferred because of the limitation of a based on data and state flowcharts. \$

to the present invention, a floating point number and an Integer are both handled at a simulation stage, and results of processing with the both are output. With this number from a result of simulation with an integer apolied thereto may be displayed so that a difference beween results of simutations with a floating point number [0012] According to yet another aspect of the presen invention, a simulation function is provided for simulat ing the data flowchart with application of a floating point numbers corresponding to a physical value, and of an floating point number is handled at a simulation stage while an integer is handled at a coding stage, according to output the simulation results. While conventionally a integer obtained by converting a floating point numbe ş

thereto and of an Integer applied thereto, re-

an integer conversion condition from a floating point number to an integer, and a result of back calculation to Further, a block symbol in the data flowchart obtain an integer from a floating point number using the nas Information on a floating point number, an Integer,

[0014] According to still another aspect of the present nvention, a priority function is provided for defining an order to execute a plurality of data flowcharts in the an efficient and appropriate control program can be resame hierarchy in the state flowchart. With this function,

omatically generated utilizing integer information of the

nteger conversion condition. Vehicle-use codes are au-

8 lowchart, and a vehicle-use code is generated in which the label is used as a variable name for the part to which he tabel is attached. When the attached label which is assily understandable for a user, a readable vehicle-use According to still another aspect of the present sired label to a selected symbol connection line in a data invention, a labeling function is provided to assign a decode can be generated. [0015]

According to yet another aspect of the present o a predetermined grouping restriction condition which corresponding to a plurality of block symbols in the data flowchart. Preferably, grouping is carried out according defines the number of black symbols to be grouped. Furher preferably, a part with a label attached thereto, of a symbol connection line is set as a grouping break. With invention, a grouping function is provided for grouping, grouping, readability of a vehicle-use code can be imin vehicle-use code generation, a plurality of processes [0016]

ş be readily embodied as C language code having been modified so as to suit to a vehicle ECU, but is not limited desirable ECU to be mounted to a vehicle, such as an pending on an ECU and vehicle-mounted devices to be Vehicle-use code of the present invention can to C code. A vehicle ECU of the present invention is a ECU, and a break control ECU. It is needless to say that control specification and vehicle specification which are angine ECU, a transmission ECU, a suspension control necessary in program development may be different de-

# BRIEF DESCRIPTION OF DRAWINGS

S Fig. 2 is a block diagram showing a structure of a Fig. 1 is a diagram showing a complete structure of a preferred embodiment of the present invention;

Fig. 3 is a data flowchart and a state flowchart indicative of control specification;

Fig. 4 is a diagram illustrating concept of integer log-

ic for a vehicle ECU;

Fig. 5 is a diagram showing an example of a part of Fig. 6 is a diagram showing structure of an integer

Fig. 7 is a diagram showing an example of an ger block applied to multiplication;

Fig. 8 is a diagram describing multiplication block Fig. 9 is a diagram showing a modified example of processing of Fig. 7;

the process of Fig. 8;

Fig. 10 is a diagram showing a support tool for in-teger logic;

Fig. 11 is a diagram showing a support tool for integer logic; Fig. 12 is a dlagram showing a support tool for in-

5

lably generated.

teger logic;

Fig. 13 is a diagram showing a support tool for in-

leger logic;

Fig. 14 is a diagram showing a support tool for in-

teger logic; Fig. 15 is a diagram showing a support tool for in-

teger logic; Fig. 16 is a diagram showing a support tool for in-teger logic;

23

Fig. 17 is a diagram describing a method for deter-

mining an operation order,

Fig. 18 is a diagram showing a priority function for

defining an operation order in the embodiment; Fig. 19 is a diagram showing C code resulting from the process of Figs. 17 and 18;

Fig. 20 Is a diagram showing example C code gen-

Fig. 21 is a diagram showing a C code generated based on the same data flowchart as that in Fig. 20, erated based on a data flowchart;

Fig. 22 is a diagram showing grouping process in C and given labeling in the embodiment;

Fig. 23 is a diagram showing variable name data base and expression database which are generated code generation;

in C code generation;

Fig. 24 is a diagram showing grouping process utilizing the database of Fig. 23;

Fig. 25 is a diagram showing an example of grouping process; and

Fig. 26 is a diagram showing an example of a display screen when the program generator monitors angine control.

# BEST MODE FOR CARRYING OUT THE INVENTION

[0019] In the following, a preferred embodiment of the present invention will be described with reference to the

[0020] Referring to Fig. 1, a program generator 1 has functions of generating and simulating a control specification, and for generating vehicle-use C code from the generated control specification.

EP 1 111 483 A1

2 a display 15. Any type of memory device other than a CD-ROM 19, such as DVD, may be used. Any type of memory device other than a hard disk 16 may be used. and the pointing device 14 (a mouse and so on), and flowchart showing a partial data flow, a state flowchart ter, having, as shown in Fig. 2, a CPU 10, a ROM 11, a RAM 12, a keyboard 13, a pointing device 14, a display 15, a hard disk 16, a communication circuit 17, and a CD-ROM drive 18. A program for achieving the respective functions of the program generator 1 is stored in either the ROM 11, the hard disk 16, or the CD-ROM 19, and executed by the CPU 10. An input/output device is not limited to a keyboard 13, a pointing device 14, and [0022] The user writes a specification incorporating his idea on the disptay 15 by operating the keyboard 13 thereupon the control specification is shown on the display 15 in the form of a data flowchart (a block diagram) and a state flowchart (a state transition chart, a state transition diagram), as shown in Figs. 3(a), 3(b), a data

gram generator 1. Specifically, a vehicle model is The user also inputs a vehicle specification by operating the keyboard 13 and the pointing device 14 so that a hypothetical vehicle model is formed in the proformed as a collection of expressions which describe [0024] In response to the user's instructions, the genlated. Specifically, as shown in Fig. 1, the vehicle model is controlled on the computer according to the control specification logic. When a simulation result is shown on the display 15, necessary debugging is applied. In a debug operation, the specification is modified and imerated specification (data and state flowcharts) is simuvarious vehicle motions and operations, and so on.

23

(0025) When the specification completes, vehicle-use C code is generated based on the complete specification in response to the user's instruction. Specifically, a file containing code written in the C language and corresponding to the data and state flowcharts is formed. A vehicle-use C code has an integer logic and a structure enabling higher speed processing using a smaller memory in size, compared to general C code (described below). A resultant vehicle-use C code is downloaded to an actual vehicle ECU 2, e.g., an engine ECU.

Conducting such a simulation before testing using an use C code, having been loaded into the ECU 2, is executed to begin simulation in a near-real environment. [0026] The vehicle model is input from the program generator 1 to the vehicle model device 3. The vehicle model device 3, including a DSP, simulates high speed interruption and other characteristics of the vehicle ECU actual vehicle reduces development period and results in a more effective product. After the vehicle model device 3 is connected to the vehicle ECU 2, the vehicle-Operation of the vehicle ECU 2 and vehicle model device 3 are monitored by the program generator 1 via the communication circuit 17, and a debug operation is ap-

improper points are amended in the computer, and the plied in the program generator 1. For example, should improper control operation be found, the cause thereof is detected using the data and state flowcharts. Then, amended logic is verified.

[0027] After debugging is completed, the vehicle ECU 2 is mounted to a vehicle 4 for actual testing. In actual testing, any defect in need of amendment due to a nonlinearity factor and so on of an actual vehicle is detected

[0028] In the following, a program generator 1 will be described in more detail. for final program amendment.

### Control Specification Input/Generation Function (Integer Logic)

ulation is carried out using the floating point number, so termined based on the simulation result. Therefore, C ing a conventional function contains a floating point Conventionally, at a control specification input stage, a chart is prepared for processing a floating point number indicative intact of a physical value. Next, a simthat acceptability of the control specification can be decode which is automatically generated from the chart us-[0028] 8

showing an entire control flow.

number) is applied to C code for a vehicle-use ECU to ence between vehicle-use C code and automatically generated general C code, and a major reason for forcsuch a demand, preferably, integer logic (a fixed point generate C code with a data bit of a different number from that of general C code. This is a significant differ-[0030] Here, vehicle control requires making best use of the capacity of the CPU of an ECU in view of minimizing costs and increasing processing speed. This demand is particularly significant in control of devices with high speed rotation, such as an engine. In order to meet ing manual generation of vehicle-use C code.

[0031] In this embodiment, an integer block (described later) is introduced so that integer logic can be cation input stage, and whereby vehicle-use C codes having an integer logic can be automatically generated taken into consideration as early as at a control specififrom a control specification.

Conversion from Floating Point Number to Integer

[0032] A floating point number is converted into an Ineger, following the below expression (1).

# x\_int=int(x\_float-OFFSET)/SLOPE

8

wherein x\_float is a floating point number, x\_int is an integer, OFFSET (or blas) and SLOPE (or LSB) are integer conversion condition.

જ

Summary of Integer Logic

according to a detected physical quantity p to an A/D vehide ECU 2. Using integer logic, the vehide ECU 2 be, e.g., an unsigned integer of 16 bits. The integer data Q is converted into a control signal to be output to an Fig. 4 illustrates the concept of integer logic in a vehicle ECU. The sensor 21 sends a voltage signal v converter 22. The A/D converter 22 converts the received voltage signal v into integer data P to send to the obtains integer data Q for a control parameter from the integer data P of a detection signal. Integer data may actuator 23, or an object to be controlled, for generation of physical quantity q.

equivalent to that which would be produced according to the original control logic with respect to a physical value. SLOPE, OFFSET at the input/output portion of the integer togic includes a plurality of operation process mediate process. These St.OPEs, OPPSETs should be set such that the integer logic is optimized in view of a In Fig. 4, the integer logic must produce a result vehicle ECU 2 are ip, lq, op, oq, respectively. In actuality, stages, and has many SLOPEs, OFFSETS in an Intermemory size and an execution time.

8

52 8 In a case wherein a control specification (logic) generated by a user includes a transfer function, as logic, the seventh item may be always caused zero when SLOPE which is suitable for the first item is applied shown in Fig. 5, and identical SLOPE is used for all seven items of the transfer function in constituting an integer to all seven items because factors differ significantly between the first and seventh items. [0035]

flow, while too large a SLOPE may result in negligence an input of a sensor. Further, setting an improper Generally, too small a SLOPE may cause over-SLOPE may result in a control logic containing a lengthy component. In order to avoid these problems, readily setting of a sultable Integer conversion condition is de-

55

### (3) Integer Block

\$

ŧ [0037] In order to generate a suitable integer logic in dition can be readily set, an integer block, shown in Fig. 6, is introduced in this embodiment. An Integer block is a data flowchart so that suitable integer conversion cona block symbol used in a data flowchart, which a user can write into a data flowchart.

types of information pieces, namely (1) integer, (2) SLOPE (LSB), (3) OFFSET (bias), (4) float (a floating As shown in Fig. 6, an integer block includes OFFSET". (6) is a back float, or a result of back calcupoint value), (5) Data Type, (6) "Integer" SLOPE [0038]

8

Function of an integer block will be described referring to Figs. 7 and 8 and using multiplication as an fation to obtain a float from an integer. (0039)

[0040] In the data flowchart of Fig. 7, input data 1, 2,

55

and SLOPE and OFFSET for each data are input to a multiplication block. The multiplication block obtains the above mentioned six types of data items from these inputs, and outputs them.

Fig. 8 shows an expression describing a process in a multiplication block. In this process, input values x, y are converted into integers using offsets 1, 2, and gers is output (integer output). Meanwhile, a float output is "xy". Comparison between expression (1) and "xy" eads to SLOPE and OFFSET on the output side (Fig. slopes 1, 2, respectively, and a product of the two inte-8, expressions (2), (3)). Further, a back float (=integer '

The user can ascertain the simulation result by referring to the display in Fig. 7. "Float" indicates whether or not basic control logic is correct. Further, "float" and "back float" are compared. When the difference (0.1) is within a tolerable range, it is known that integer ence is not within the tolerable range, it is known that nappropriate integer logic, or integer conversion condition, namely "slopes 1, 2", is set. Then, "slopes 1, 2" are repeatedly adjusted until an appropriate result is oblogic is appropriate. On the other hand, when the differ-SLOPE<sub>OUT</sub> + OFFSET<sub>OUT</sub>) is also output. [0042] tained.

ification. In actuality, control specification constitutes of a series of numerous connected blocks. In this embodment, "SLOPE" can be verified and adjusted in each block in a control process. For example, when a significant difference is found between "float" and "integer" in a certain block, integer conversion condition is adjusted [0043] Figs. 7 and 8 show only a part of a control specin any block preceding the focused block

[0044] With the above arrangement, "float" and "integer" can have substantially equivalent meaning over the entire control process. In other words, a situation with a difference between "float" and "integer" becoming larger as it goes to latter blocks, can be avoided. Also, a situation in which input of a certain sensor is ignored because of an improper SLOPE set, can be avoided

[0045] As shown in the third expression in Fig. 8, OFF-SET on an output side includes input values x, y. When the influence thereof is not negligible, the user can chose modified process, as shown in Fig. 9. Specifically, "offset/slope" is added to an integer prior to the multiplication whereby the item for "offset" on the output side becomes zero, as is shown in the first expression in Fig.

[0046] Note that, although multiplication is used as an Figs. 10 to 16 show various integer blocks which support Integer logic. Although information y(5) to y(8) are added to an integer block in this tool, as shown in Fig. 10, basic information y(1) to y(4) is the same as described above. example in the above to describe a function of an integer block, similar function can be set to other operations Also, LSB in Figs. 10 to 16 corresponds to "slape".

[0047] As described above, in this embodiment, as a result of introduction of an integer block, a control specification in the form of a chart expresses an integer logic.

EP 1 111 483 A1

As sultable integer conversion condition can be set through comparison between theoretical control logic be easily made. The complete control specification itself (real number operation) and Integer logic (integer operation), suitable integer logic including fewer bugs can is usable for a computer simulation. In addition, vehicleuse C code can be automatically generated by extractng an integer logic part from the specification. As the control specification and vehicle-use C code have parts in common, bug generation is reduced at a code generation stage, and program verification, such as debugging, can be easily applied.

# Priority Function (Regulation of Operation Order)

each module or operation expression is made into a triggered subsystem and given a function call from a state 0048] For determination of an order to execute a plurality of modules or operational expressions, generally,

user imparts a number indicative of a priority order to expression is made as a prioritized subsystem same hierarchy can be easily determined. In addition, lionally provided, as shown in Fig.18. Specifically, the the data flowchart, whereby each module or operation (1\_Prior\_Subsystem, 2\_Prior\_Subsystem). Operations [0050] The priority function leads to an advantage that an order to execute operations in data flowcharts in the another advantage can be achieved that a C cod which [0049] In this embodiment, a priority function is addiare executed in the order of names of the subsystems. s easier than conventional one can be generated. This is preferable in view of readability and a memory capacllowchart, as shown in Fig. 17.

## C Code Generation (Labeling)

chart has integer information. Therefore, a series of [0051] C code generation is applied in response to a plication of an integer block, each block in a data flowlions. By generating integer operations, vehicle-use C referring to the example in Fig. 7, integer operations (two nteger inputs and one integer output) are extracted, at a chart level, from information groups to be input or outuser's instruction. As described above, as a result of apblock groups can be processed using integer operacode having an integer logic are generated. Specifically, put with respect to a multiplication block.

8 for the above use, a tool for automatically producing a general C code based on a chart and a modeling tool nto which a C code is written are prepared, and the processing which is absent from a general C code and [0052] In order to modify an extent software product node! is modified so as to suit to the integer logic. Then, unique to a dedicated command, and a dedicated comnand resulting from a modified general C code are also

Fig. 20 shows an example of a C code gener-0053]

ated from a data flowchart. According to a general C code generation method, computer-determined varia-This results in a difficult to read C code ble names are given to the respective parts, as shown in the drawing.

gram developer must read a C code in testing using an actual ECU, which is conducted for motion verification in vehicle system development. Therefore, user friendliness is desired in automatic vehicle-use C code gensideration in a general C code automatic generation method, which is mainly concerned with coincidence between a C code and a chart. Nevertheless, often a pro-[0054] That is, readability is not fully taken into con unfriendly to human users. eration.

attached to the selected connection line. In Fig. 21, as an example, labels 'l\_x', 'l\_y', 'l\_z', 'l\_xyz' are altached. In actuality, more easily understandable labels, shown in Fig. 21, is employed to improve readability. A user clicks a connection line in a block symbol in the data flowchart, in response to which a desired label is such as a module name, a sensor name, and so on, are preferably attached. In C code generation, the label may [0055] In this embodiment, a labeling function, as be used as a variable name of the part with that label attached. As a result, easily readable C codes are generated, as shown in Fig. 21. 5

in Fig. 21, as the former may help easy reading of C Note that labels may be attached to only a desired line, rather than to all connection lines, as shown codes. As for a line without labeling, a computer-deter-[90026] 8

### [0057] 35

and an expression for introducing one variable is generated corresponding to each block. Therefore, resulting C code contains numerous variables and In general C code generation, a different variable name is given to each block in a data flowche expressions, making it long and less readable.

\$

ing between input and output. With C code on the left side without grouping, two expressions are gencode on the right side with grouping, operations for erated according to the number of blocks. With C two blocks are integrated into a single expression, reading a C code are deleted, and easily readable In order to address this problem, in this embodiment, a grouping function is employed to improve readability. In Fig. 22, there are two blocks intervenin which variable names unnecessary for a user C code is thereby generated.

may result in a lengthy expression with poor read-Here, grouping of too large a number of blocks Ining the number of blocks contained in one group s preferably determined. In this example, the maxability. To address this problem, a condition for de-

mined variable names may be used intact.

## Grouping in C Code Generation

mum number of such blocks is set at two. That is, for one expression of a C code, operations will be

5 Grouping process will next be described with ID is attached to each block in a data flowchart for use as a variable name. Then, a variable name (signal name) database is made for correlating ID and a user-designated label (Fig. 23 (a)). When no correference to Figs, 23 and 24. In C code generation, responding label can be found, ID is used intact.

nal is determined with reference to an expression database (Fig. 23 (b)) (S12) and when the result is Is determined that the condition is held, a variable name database is then searched (\$14) to employ a corresponding label as a variable name. When no attached label is detected, an ID is used. In Fig. 24, ID of an input signal to a focused negative (less than twice), the expression database is searched (S13) to adopt an expression corresponding to an input ID. On the other hand, when it block is detected, using a function to detect an input signal (S11). Then, whether or not grouping condition (twice at maximum) is held with each input sig-

able name employed at S14, an expression for the focused block is generated (S15). Whether or not a the expression generated at S15 is registered to the expression database for preparation for the process in the next block (S17). On the other hand, when the condition is held, the expression generated at Using an expression employed at S13 or a vargrouping condition is held with respect to the generated expression is detected (S16). When it is not, S15 is written into a file (S18). In Fig. 22, an expression for two blocks combined is output.

With reference to Fig. 25, a specific example of to obtain an expression (x+1). As the number of grouping for this expression is only once (i.e., the (x+1) is used as an input signal. Then, a search is carried out with respect to an input signal S2. As the number of grouping for an input signal S2 is twice, s variable name database is then searched to adopt abel x2 as an input signal. Therefore, an operation base is searched with respect to an input signal S1 above mentioned condition is held), the expression within the focused block will be "(x+1)"x2" (multipli-With input signals S1, S2, an expression datathe above grouping processing will be described

55 Next, an output signal t\_x is retrieved from the variable name database. As two groupings have the grouping condition is held. Then, t\_x=(x+1)\*x2 s output. If grouping condition is still to be held at his stage, the generated expression is registered been conducted with respect to the input signal S1,

In grouping processing, a part in a chart, where a label is attached is always determined as a group-

When labels attached to an appropriate part are assigned from a chart prepared by a human user, C code having a structure corresponding to that of the chart is generated. With this processing, more eastained in an operation expression for a C code. ing break. All labels attached by the user are conily readable C code is automatically generated. In an example wherein a user attaches a label to the beginning and end of a data flowchart, very readable C code is automatically generated, which has definite variables at the beginning and end thereof and

As described above, according to this embod-Iment, when grouping is applied in C code generation, readability of an automatically generated C code can be improved, and memory can be conserved through reprocess segments in appropriate intervals. duction of variable names.

## Download to Vehicle ECU

[0060] As described above with reference to Fig. 1, the program generator 1 downloads an automatically generated vehicle-use C codes to the vehicle ECU 2 in response to a user's instruction. The vehicle ECU 2 then executes the downloaded vehicle-use C code to control the vehicle model device 3 (DSP). The program generator 1, monitoring the control being executed, displays the execution state in an appropriate format on the display. Fig. 26 shows an example of a monitor screen when the vehide ECU 2 is an engine ECU. The user operates the program generator to conduct motion verification and a debug operation with respect to a vehicleuse C code. 52

[0061] In this embodiment, a suitable vehicle-use C code having few bugs is generated. The vehicle-use C code is in common with the original control logic. The vehicle-use C code has higher readability. Therefore, a debug operation can be easily and efficiently performed. [0062] As described above, according to the present invention, as a vehicle-use code corresponding to the control specification input by the user is automatically generated, the number of coding operation steps can be significantly reduced. The input control specification Itself is usable for computer simulation, as well as usable in a vehicle-ECU after conversion into a vehicle-use code. This enables significant reduction of the number of debug operation steps. As a result, the number of steps necessary to ensure reliability can be reduced, as well as a development period and costs. The advantage of the present invention for assisting a process from preparation of a specification to logic debugging, is remarkable particularly in a situation with overgrown and complicated vehicle control programs. \$ ŧ 8 33

[0063] Also, according to the present invention, a highly readable vehicle-use code can be automatically generated. As a result, the number of steps for a debug operation and so on can be further reduced.

EP 1 111 483 A1

5

## NDUSTRIAL APPLICABILITY

cording to the present invention can be utilized in development of a vehicle control program to be executed by [0064] As described above, a method and apparatus for assisting development of a vehicle-use program aca vehicle ECU.

### Claims

9

 A method for assisting development of a program for a vehicle, comprising: 5 a program generation step of generating a vehicle control program using a program generaof vehicle-use code based on a control specifitor having a function for generating a segmen

a downloading step of downloading the generated vehicle control program to a vehicle ECU; cation Input;

8

debug step of debugging the vehicle control program by causing the vehicle ECU to execute the vehicle control program.

22

A method for assisting development of a program for a vehicle according to claim 1, wherein

3 the program generator which inspects a result of exdebugging at the debug step is carried out in ecution of the vehicle control program by the vehicle

33 for a vehicle according to claim 2, further compris-A method for assisting development of a program

ŝ hicle ECU to a vehicle model device which models a vehicle to be controlled, to cause the a control execution step of connecting the vevehide ECU to control the vehide model derice; and

an inspecting step of inspecting the vehicle ECU and the vehicle model device while control A method for assisting development of a program for a vehicle according to claim 3, further compris8

s model generation step of generating a vehicle model in the program generator based on a vehicle specification input; and

S hicle model generated at the model generation a model download step of downloading the vestep to the vehicle model device.

A method for assisting development of a program

wherein the segment of vehicle-use code is a segfor a vehicle according to any one of claims 1 to 4, ment of general code modified to suit an integer logic to be processed in the vehicle ECU.

for a vehicle according to claim 5, wherein a vehicle control program is generated at the program gen-A method for assisting development of a program eration step, so as to accord to an integer conver sion condition input into the program generator.

A device for assisting development of a program for a vehicle, comprising: ۲.

input means for inputting a vehicle control specilcation;

download means for downloading the vehicle program generation means for generating a segment of vehicle-use code for a vehicle concontrol program to an external vehicle ECU; trol program based on the control specification;

tion of the vehicle control program by the vehioutput means for outputting a result of execuA device for assisting development of a program for a vehicle, comprising: æ

ta flowchart and a state flowchart indicative of a chart generation function for generating a da-

a program code generation function for generating, based on the charts generated, a segment of vehicle-use code for a vehicle control program having an integer logic to be proca vehicle control specification; and essed by a vehicle ECU.

obtained by converting a floating point number, to output results of simulations with floating point A device for assisting development of a program for flowchart with application of a floating point number corresponding to a physical value and of an integer a vehicle according to claim 8, further comprising: a simulation function for simulating the date æ

number applied thereto and of the integer applied

thereto, respectively.

A device for assisting development of a program for a vehicle according to claim 9, wherein a result of back calculation to obtain a floating point number from a result of simulation with an integer applied thereto is displayed so that a difference between results of simulations with the floating point number applied thereto and of the Integer applied thereto, respectively, can be determined. A device for assisting development of a program for

- A device for assisting development of a program for a vehicle according to claim 11, wherein the integer conversion condition is able to be adjusted based on a result of the simulation ĕ
- a vehicle according to any one of claims 8 to 12, A device for assisting development of a program for further comprising
- a priority function for defining an order for executing a plurality of data flowcharts in a same hierarchy in the sate flowchart.

ating, based on the generated chart, a segment of vehicle-use code for a vehicle control program having an integer logic to be processed by a vehicle ECU.

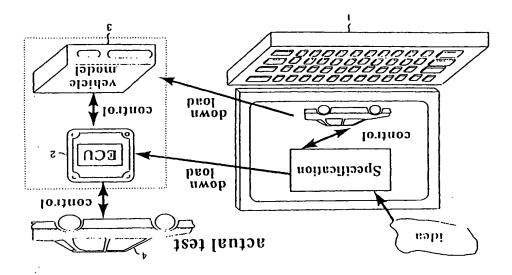
- 14. A device for assisting development of a program for a vehicle according to any one of claims B to 13,
- a labeling function for assigning a desired label to a symbol connection line desirably selected
  - in the data flowchart,
- a vehicle-use code using the label as a variable name of a part with the label attached is gen-
- a vehicle according to any one of claims 8 to 14, A device for assisting development of a program for further comprising
- a grouping function for grouping a plurality of processes corresponding to a plurality of block symbols in the data flowchart when the vehicle-use code is generated.
- A device for assisting development of a program for a vehicle according to claim 15, wherein grouping is applied according to a predetermined grouping restriction condition which defines a number block symbols to be grouped.
- A device for assisting development of a program for a vehicle according to claim 15 or 16, further com-
- to a symbol connection line desirably selected a labeling function for assigning a desired label in the data flowchart,
- a part with the label attached thereto is set as
- 18. A device for assisting development of a program for

wherein a segment of vehicle-use C code is generated by modifying, using the program code generation function, a segment of general C code to suit a vehicle ECU. a vehicle according to any one of claims 8 to 17,

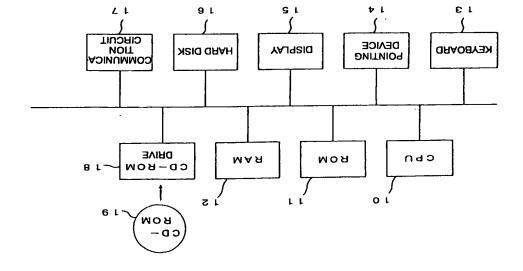
- 19. A computer readable memory medium for use in assisting development of a program for a vehicle,
- a chart generation function for generating a data flowchart and a state flowchart indicative of ı program code generation function for generbearing a program causing a computer to execute a vehicle control specification, and

EP 1 111 483 A1

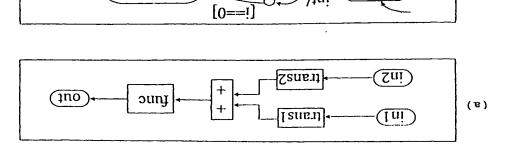
### Fig. 1











(ESV)

EEI

/ເມລາ )

[!<n]{!++!}(u>[]

**Jini** 

(to ())(o<))

**(**9)

7

Ξ

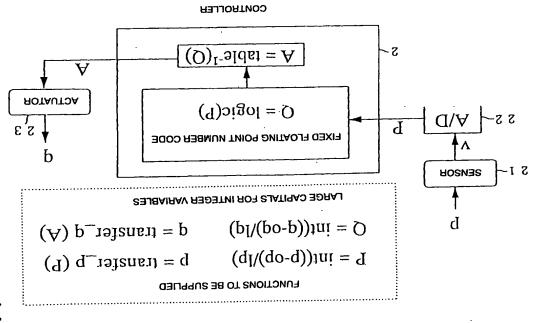
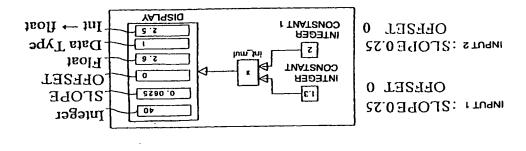


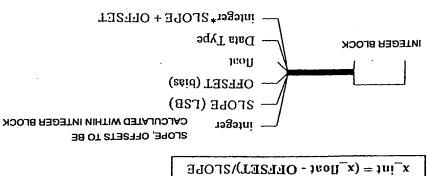
Fig. 5

$$y(k) = 0.7888 \ y(k-1) + 0.1784 \ y(k-2) - 0.1000 \ y(k-3) - 0.0010 \ u(k) + 0.0150 \ u(k-1) - 0.0040 \ u(k-2) - 0.0020 \ u(k-3)$$
EXAMPLE OF AN EXPRESSION INCLUDING A FLOATING POINT NUMBER



9

5



CONVERSION EXPRESSION FLOATING POINT NUMBER → INTEGER

(I)

MULTIPLICATION

$$\frac{10401}{(x - offsetl)} \times \frac{(y - offset2)}{slope2}$$

(2)

MULTIPLICATION

 $\frac{\text{offest1}}{\text{slope2}} \times \left( \frac{\text{slope2}}{\text{slope2}} \right)$  $+\frac{\text{offset2}}{\text{slope2}}$ τυστυο

slope 1 \*slope2

DESIGNATED STOPE:  $slope_0 \rightarrow output$ 

**₽** 

OPFI  $_{int}$ : offset Uslopel, OFF2  $_{int}$ : offset 2 slope2 DESIGNATED OFFSET: offset 0→ output

 $((X_{ini} + OFF1_{ini})^*(Y_{ini} + OFF1_{ini})^*slope l *slope l *slope o free o) slope o free o$ NULTIPLIED INTEGER VALUE:

```
Fig. 10
```

```
(8) 1D TABLE LOOK UP
                                                 THIRS (T)
                                              SUPPLUS (6)
                                               NOISINIO (5)
                                         (1) MULTIPLICATION
           (12) INTEGER SCOPE
               VALIBIOT THU (11) HERMUN THIOR NOTITION (E)
                                (2) CONVERSION TO FLOATING
                   TU92 (01)
                                 (1) CONVERSION TO INTEGER
         (9) SO TABLE LOOK UP
                ( S ) INTEGER LOGIC SUPPORT TOOL HAS BLOCKS BELOW
                                  y(4) = floating point value
           \lambda(8) = NEGATIVE
                                           \lambda(3) = OLESEL
                y(7) = ZERO
                                               \lambda(S) = \Gamma S B
               \lambda(0) = CARRY
                                        y(1) = integer value
      y(5) = signed or unsigned
      OF THE BELOW
(L)
```

Fig. 11

```
KERMUN TO FLOATING POINT NUMBER (S)
                                                      Suley mumixem (S)
                                                       Juley muminim (1)
                                               DIALOGUE BOX DISPLAY
                                     10 tucket
          y(8) = NEGATIVE
                \gamma(7) = ZERO
                                                       T32710 (4)
   \gamma(5) = \text{signed or unsigned}

\gamma(6) = \text{CARRY}
                                             (2) zigned or unsigned
   y(4) = floating point value
                                                      dignol tid (1)
         y(1) = integer value
y(3) = 1.513
y(3) = integer value
y(4) = integer value
                                            DIALOGUE BOX INPUT
                                                                (1) соилензіон то інтесен
(2)
```

2

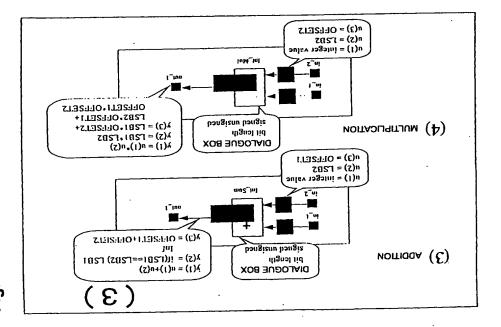


FIG. 13

γ(1) = u(1)%u(2) γ(2) = LSB1 γ(3) = OFFSET1/u(2)

\(\lambda(1) = \(\overline{1}\)\(\text{1}\)\(\text{1}\)\(\text{2}\)\(\text{1}\)\(\text{1}\)\(\text{2}\)\(\text{1}\

(7)

hd\_Div

u(1) = integer value u(2) = 1,5132 u(3) = OFFSETT

u(1) = integer value u(2) = 1,5B2 u(3) = 0FF5EF1

surprus (6)

Noisinia (5)

22

Fig. 14

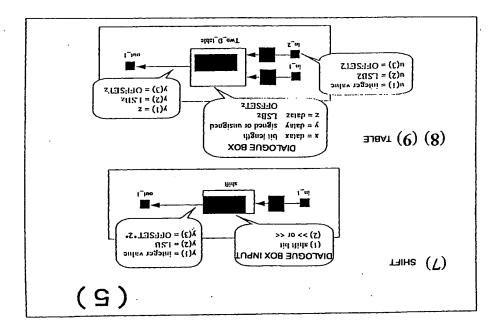
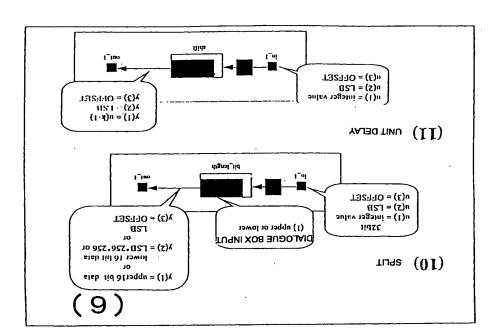


Fig. 15



7

EP 1 111 483 A1



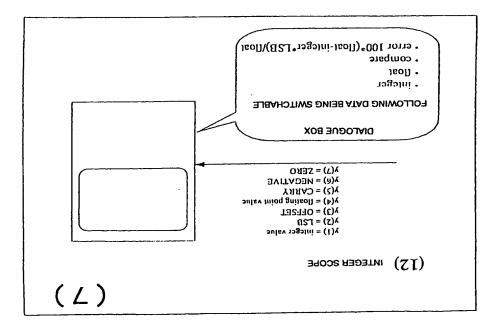
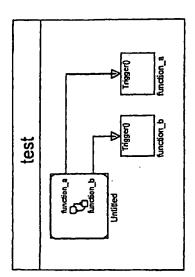
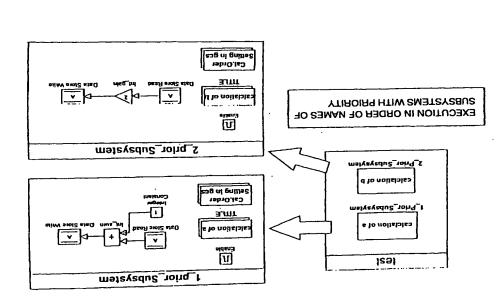


Fig. 1.



76



Y = Y \* Y

:I+A=A

(biov) test biov

test.c PRESENT INVENTION

 $\forall \star \nabla = \nabla$ 

void function\_b(void)

(void function a (void) c.test

function\_b();

function\_a();

void untilled (void)

CONVENTIONAL

Fig. 20

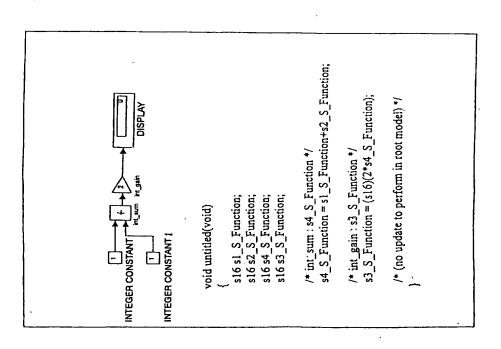
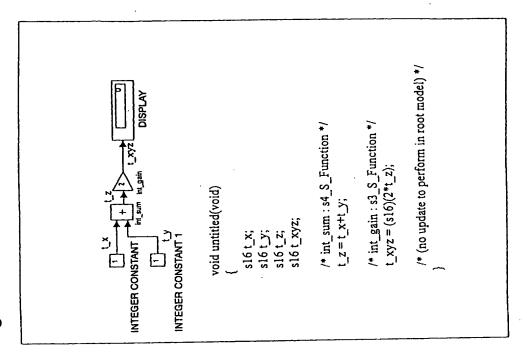


Fig. 21



```
s4\_S\_Function = t\_x+t\_y;

t\_xyz = (s16)(2*s4\_S\_Function);
/* int_gain : $3_5_Function */
(_xyz = ($16)(2*t_x+t_y);
                                                                 $16 (_y;
$16 (_y;
$16 (_y;
                         2/x_191s
1/191s
2/x_191s
                                                                                191s
                (biov)ballilan biov
                                                                    (biov)baltitnu biov
                    ВВОПЫИВ
                                                               эмічиола тионтім
                                                           INTEGER CONSTANT 1
                    YAJ92IQ
                                                             тилебей соизтаит
```

GROUPING

Signal Label

(a)

EXPRESSION

s1\_S\_Function

(a)

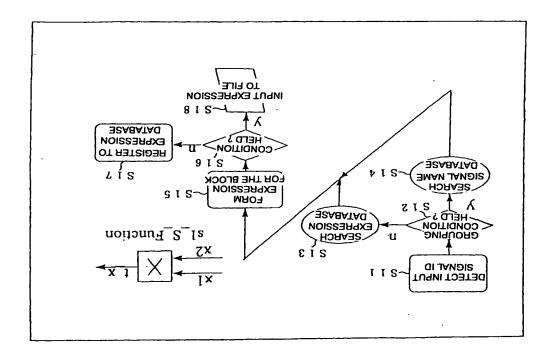


Fig. 25

EP 1 111 483 A1

IG WITHIN

PROCESSING WITHIN

MULTIPLICATION BLOCK (\$3 S Function)

IS S Function x S S Function

s2\_S\_Function s3\_S\_Eunction

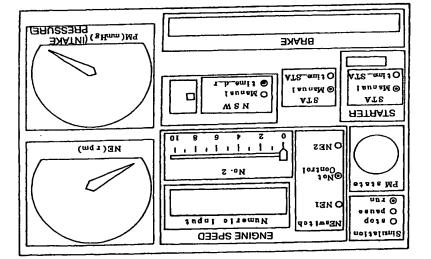
**32ABATAO 3MAN 3JBAIRAV** 

EXPRESSION DATABASE

I+X

sl\_S\_Function

\$



INTERNATIONAL SEARCH REPORT

A CLASSIFICATION OF SUBJECT MATTER Int.Cl* 605B19/02		
Azondring to International Patent Clearification (IPC) or to both national clearification and IPC. B. FIRST DS SPARCHED.	ational classification and IPC	
1.6	l by classification symbols)	
Documentation control other than minimum decementation to the extent that such decements us included in the fields wented Jitsaupo Shilman Koho 1936-1998 Toroku Jitsaupo Shilman Koho 1996-1998 Kokai Jitsaupo Shilman Koho 1996-1998	ne extent that such documents are included Toroku Jitsayo Shihaan Koho Jitsayo Shihan Toroku Koho	in the fields searched 1994-1998 1996-1998
Bedronie dats bure omwilted during ibs international sourch (tama of dats bure and, where predicable, tourch terms med)	me of dete buse and, where predicable, soo	arch terms used)
C DOCUMENTS CONSIDERED TO BE RELEVANT		
ation of ducument, wi	spropriate, of the relevant passages	Relevant to claim No.
Y JP, 7-93137, A (Hitachi, Ltd.) 7 April, 1995 (07. 04. 95) (F)	(Family: none)	1-2, 5-8,
Y JP, 1-319880, A (NEC Corp.) 26 December, 1989 (26. 12. 8	), 89) (Family: none)	1-2, 5-8, 18-19
Further documents are listed in the continuation of Box C.	See petent family annex.	
*** Second diving the control of the december of the control diving the proof and of the control diving the proof and of the or with it and control diving the proof and of the or with it as online denotes the published on of the december of the control of the c	The transcription of the districtions filling the support of the s	tions filled the orphrish estimated to externand estimated investors cannot be to investor as investors table for the decreased in the bed document in external, well combines as commit, well combines
the priority date chalmed	decorates sweather of the same patent family	nily
Due of the actual completion of the international search 14 July, 1999 (14. 07. 99)	Date of mailing of the international search report 27 July, 1999 (27. 07. 5	ch report 07.99)
Name and multing address of the ISA Japanese Patent Office	Authorized officer	
Facinite No.	Talephane No.	
East DOT/10 A / 10 (second sheet) (luly 1002)		